Documentation and Reporting: Expense Tracker Integration Process

This document outlines the integration process of the Expense Tracker system, explaining the methods used, system setup, challenges faced, and how they were solved. It provides an overview of the steps taken to combine different components of the system and ensure everything works smoothly. The document also discusses the technical setup, including how the backend, frontend, and database work together.

System Configurations

1. Development Environment:

- **Programming Language:** Java & Python
- **Integrated Development Environment (IDE):** Netbeans & Pycharm
- **Database:** MySQL
    - **Pycharm version:** 3.1.1
    - **Netbeans version:** 8.2
    - **Python version:** 3.13.1
    - **MySQL Version:** 8.0

2. Integration Setup:

- **API Communication:** RESTful API Communication (Representational State Transfer):
- Uses HTTP methods (GET, POST, PUT, DELETE) perform operations on resources (such as expenses, categories, or users).

Integration Methodologies

- Java & Python connected to the API using Pycharm(Python) importing Flask as Python Interpreter.

Error Handling in MySQL:

Problem:

error#1: Cannot establish a connection to jdbc:mysql://localhost:3306/studentdb?zeroDateTimeBehavior=convertToNull using com.mysql.jdbc.Driver (Unable to load authentication plugin 'caching_sha2_password'.)

Solution:
ALTER USER 'YOUR_ROOT_USER_NAME'@'localhost' IDENTIFIED WITH mysql_native_password BY 'YOUR_ROOT_USER_PASSWORD';

Challenges and Solutions

1. API Communication Challenges

Challenge:

API calls between the frontend (Java) and backend (Python) face issues like data inconsistency, incorrect formatting, or failed requests. These issues can affect the flow of expense data or transactions.

Solution:

Implements standardized API design using RESTful APIs with consistent data formats (JSON) for communication between frontend and backend.

2. MySQL Database

Challenge:

The application is unable to establish a connection to the MySQL database because of an authentication plugin mismatch.

Solution:

Change MySQL User Authentication Plugin

3. Handling Multiple Data Formats

Challenge:

The frontend (Java) and backend (Python) need to handle different types of data (e.g., strings, integers, dates) that need to be formatted properly for storage and display.

Solution:

Standardize the data format for consistency across the system.

Integration Success for Expense Tracker

The integration of the Expense Tracker application, utilizing API communication, MySQL as the database, Java for the frontend, and Python for the backend, has been successfully implemented. The system now allows seamless interaction between the client (Java-based frontend) and the server (Python-based backend)

API Communication:

RESTful APIs developed using Python (Flask/Django) have been successfully integrated to handle CRUD (Create, Read, Update, Delete) operations for expense data.

The frontend (Java) communicates with the backend using HTTP requests, sending and receiving data in JSON format, ensuring flexibility and ease of integration.

Database Integration:

The MySQL database is now fully integrated with the backend to store user data, expense entries, and categories.

The backend successfully interacts with the database using SQL queries to add, retrieve, update, and delete expenses in real time.

Database Connectivity:

Successfully connected the MySQL database with the Python backend using the MySQL JConnector.

Future Recommendations

- Enhance the user interface based on feedback to further improve usability.

The Expense Tracker system has been successfully integrated, and all components (frontend, backend, database) are functioning smoothly. Users can now easily track and categorize their expenses in a secure, real-time environment. The application is stable, secure, and ready for use, with database queries and efficient API communication ensuring a smooth user experience.